Olivier Mengué – Code & rando

« Formule Excel : extraire le nom de fichier d'un - Je vais au FOSDEM 2008 »

Les programmes télé de Télérama en XMLTV

Par Olivier Mengué le lundi 11 février 2008, 00:19 - XMLTV - Lien permanent

curl EPG grille télé HTML JavaScript Perl programme télé service web TV télé Télérama webdev webservice XHTML XML XMLTV XSLT xsltproc

XMLTV est un format informatique pour les programmes de télévision. Ce format est utilisable dans un nombre grandissant d'applications. Le magazine Télérama propose sur son site une grille des programmes télé, mais juste consultable sur le web. Pas de XMLTV.

Je vous propose donc de combler ce manque avec un petit cours de reverse engineering sur le web et la démonstration de quelques outils pour arriver rapidement à vos fins. La grille de Télérama est une excellente cible pour ce genre d'exercice parce que, vous le verrez, la tâche n'est pas simple, mais possible. Elle donne l'ocassion de montrer l'usage de plusieurs outils (curl, XSLT, Perl) et c'est un exemple de ma démarche de prototypage rapide en utilisant le meilleur de chacun.

MàJ 2008-02-15 : j'ai profondément remanié l'introduction et la conclusion suite à quelques commentaires de lecteurs.

Cette grille fonctionne avec des services web W. Cela se voit en cliquant sur les flèches à gauche ou à droite du bandeau des heures : seule la grille est rechargée, et non pas la page entière. Un service web est par définition exploitable pour récupérer des données structurées, plus légères que la page entière affichée dans le navigateur.

La première étape : un survol rapide du source HTML de la page. On repère les éléments suivants :

1. La page est très grosse : 208 381 octets. Toute la grille initiale est donc fournie en HTML. Ce sont les mises à jour qui sont fournies en service web.
2. La page est déclarée comme XHTML W d'après l'entête :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
xml:lang="fr"lang="fr">
```

3. Plusieurs balises <SCRIPT> en majuscules. Or, en XHTML ces balises doivent être en minuscules. Ceci indique déjà que malgré la déclaration le XHTML n'est pas valide et donc ne sera pas utilisable directement dans un analyseur XML.
4. Beaucoup de références à des fichiers JavaScript additionnels. Les familiers d'Ajaxian.com détectent vite lesquels sont des librairies JavaScript bien connues. Pour les autres une petite recherche sur Google sur le nom du fichier donne rapidement la réponse.
5. Un bloc de JavaScript révélant l'adresse du service web et le framework utilisé pour l'appeler :

```
<scripttype="text/javascript">
var
xajaxRequestUri="http://television.telerama.fr/tele/grille.
```

Catégories

- Général
- Rando
- Code
- Vélib'
- curlIE
- **XMLTV**

Rechercher

OK

Accueil - Archives

Tags

carte code France HTML Linux Maemo Mapper metadata mobile N800 Nokia N800 Paris Perl Python Ubuntu video Vélib' webdev Windows XMLTV XSLT

Tous les tags

S'abonner

Fil des billets

Fil des commentaires

```
php";
var xajaxDebug=false;
var xajaxStatusMessages=true;
var xajaxWaitCursor=true;
var xajaxDefinedGet=0;
var xajaxDefinedPost=1;
function xajax_chargerProgramme(){return
xajax.call("chargerProgramme", arguments, 1);}
</script>
```

6. Des champs de formulaire tel que celui-ci :

```
<selectstyle="width:
120px;"onchange="ajouterChaine(this.options[this.options.se
lectedIndex].value);">
<optionvalue="">Ajoutez une cha&icirc;ne</option>
<OPTGROUPlabel="Groupes de cha&icirc;nes"/>
<optionvalue="192,4,80,34,47,111,118">Chaînes
hertziennes</option>
<optionvalue="192,4,80,34,47,111,118,445,119,195,446,444,78
,234,481,226,458,482">Chaînes TNT</option>
```

On voit qu'il s'agit d'une liste de codes pour les différentes chaînes de télé à afficher.

7. Des appels de fonction JavaScript :

```
<ahref="javascript:void(0);"onclick="return
changerJour(0);">Aujourd'hui</a>
<ahref="javascript:void(0);"onclick="return
changerMaintenant(0);">Maintenant</a>
<ahref="javascript:void(0);"onclick="return
changerCeSoir(0);">Ce soir</a>
```

On note ainsi quelques noms de fonctions qu'on pourra chercher plus tard.

8. Le code HTML correspondant aux éléments de la grille. Par exemple :

```
<divclass="emission
magazine"style="width:16px;left:677px;background-image:
url('http://icon-
telerama.sdv.fr/iconsv2/bg_grille_genre_vide.gif');height:5
1px; cursor: pointer; z-index:
10;"id="emission_8069449"onclick="return
afficherEmission('8069449', '226');"alt="Journal de la nuit
- Vendredi 08 février de 02h30 à 02h34"title="Journal de la
nuit - Vendredi 08 février de 02h30 à 02h34">
<divclass="conteneur">
<spanclass="genre"></span><br />
<spanclass="titre">...</span><br />
<spanstyle="resume"></span>
<divid="data_8069449"style="display:none;">{"Id_Diffusion":
"8069449","Id_Emission":"9648923","Id_Chaine":"226","Date_D
ebut":"2008-02-08 02:30:00","Date_Fin":"2008-02-08
02:34:00","Titre":"Journal de la
nuit","Sous_Titre":"","ShowViewFr":"46271175","note_T":"0",
"Id_Rubrique":"8108","Rubrique_Libelle":"Journal","Rubrique
_Niveau":"3","Type":"Magazine","Chaine_Nom":"i
Télé","Logo":"226.gif","Id_Hierarchie":"040801","DureeEnSec
ondes":"240","resume_court":"","resume_long":"","dateheurec
haine":"Vendredi 08 février de 02h30 à 02h34 sur i
Télé","intervenant":""}</div>
</div>
```

On remarque en particulier l'élement `data_8069449` qui n'est pas affiché et qui ressemble à des données au format JSON W.

L'étape suivante consiste à récupérer une copie de la page et de tous les scripts additionnels, feuilles de style et images. Il suffit dans Firefox de choisir le menu Fichier → Enregistrer sous... → Page complète.

On obtient les fichiers suivants :

```
$ ls
grille.php.html grille.php_fichiers/
$ ls grille.php_fichiers/
1520033938Top1Bottom2x01x10.js mochikit_packed_tele_min.js
AC_RunActiveContent_min.js notes.js
ajax-loader.gif oas.js
alerte.js ong_montel_off.gif
builder.js ong_podcast_off.gif
carrousel_min.js ong_telerama_on.gif
controls.js ong_wiz_off.gif
couverture.gif pied.js
dragdrop.js prototype.js
effects.js rss.gif
empty.gif scriptaculous.js
```

```
favoris.js scriptaculous_packed_tele_min.js
fleche_droite_bottom1.gif scripts_2007.js
fleche_gauche_bottom1.gif slider.js
grille_croix.gif sound.js
grille_heures_fleche_dro.gif styles_2007.css
grille_heures_fleche_gau.gif T1_p.gif
heureFloatEntete.js T2_p.gif
hit.gif T3_p.gif
horsserie.js T5_p.gif
js_tra_authentification.html trans.gif
M5447.jpg tra_tele_v2.js
M5456.jpg urchin.js
M5495.jpg xajax.js
menu.js xtroi.js
milieu.js
$ du -b -c grille.php.html grille.php_fichiers/
210394 grille.php.html
546599 grille.php_fichiers/
756993 total
```

On peut alors les trier en trois catégories :

1. les librairies : Scriptaculous, Prototype, Xajax, MochiKit. Repérer les librairies permet d'aller sur le web chercher la doc des fonctions et des objets et d'éviter de plonger le nez dans du JavaScript compressé.
2. les pubs et les pisteurs : on les reconnaît au faible nombre de lignes et à la piètre qualité du code écrit une fois il y a dix ans et jamais mis à jour en fonction des nouvelles normes (XHTML, DOM). Exemple : oas.js
3. les scripts spécifiques au site : non compressé, avec un nom de fichier, des commentaires ou des noms de fonctions en français, ou contenant des noms de fonctions repérés dans des gestionnaires d'évènements attachés à un tag HTML (par exemple, `changerMaintenant()` vu plus haut). Chercher le nom du site (`grep telerama *.js`) permet aussi de les isoler. Ce sont bien sûr les plus importants puisque c'est là que l'on trouve les adresses des services web et la manière de les appeler.

Ce premier aperçu permet de se donner une idée de la compétence du/des développeurs. En l'occurence :

- Malgré la déclaration de l'entête, la page n'est pas du XHTML valide. La validation avec l'outil du W3C révèle ainsi 384 erreurs !
- Le fichier HTML est énorme (200 Ko), le nombre de fichiers également. Le poids total de la page est de plus de 800 Ko ! On a beau être à l'heure de l'ADSL en France, ce n'est pas une raison.
- De nombreuses librairies JavaScript sont utilisées. Leurs fonctions se recouvrent et sont redondantes. Il existe au moins 3 façons différentes de faire un appel AJAX. Est-ce que plusieurs développeurs ont travaillé sur cette page successivement ? En tout cas le manque de coordination est flagrant.
- Les tags XHTML sont détournés pour stocker des données JSON contenant elles-même du HTML (et non du XHTML). Pas étonnant que cela ne valide pas !
- Beaucoup de code mort laissé en commentaires dans le JavaScript.

En somme, pas brillant.

Nous avons repéré plus haut l'adresse d'un service web. Il est temps de le tester, en dehors du navigateur pour examiner les données retournées. Ce service est appellé avec Xajax. En fouinant dans xajax.js, on trouve vite (cherchez "postData") la façon dont Xajax envoie les données au serveur. On peut alors reconstruire une requête avec curl :

```
$ curl --data "xajax=chargerProgramme&xajaxr=$(date
+%s)&xajaxargs[]=$(date '+%Y-%m-%d
%H:00:00')&xajaxargs[]=192,4,80,34,47,111,118"
http://television.telerama.fr/tele/grille.php
```

On obtient cette fois du beau XML valide dont voici la structure :

```
<xjx><cmdn="as"t="leprogramme"p="innerHTML"><![CDATA[...]]><
/cmd>...</xjx>
```

Le contenu de `leprogramme` est le même que ce que l'on trouve dans la page HTML de base. On retrouve notamment la description des programmes à la fois en HTML et en JSON. Ce n'est pas vraiment efficace en terme de quantité de données transférées sur le réseau. Les données JSON aurait été suffisantes, et je l'avoue, cela m'aurait facilité la tâche.

Il suffit d'appliquer le filtre XSLT suivant (leprogramme.xslt) pour extraire le contenu de `leprogramme` :

```
<?xmlversion="1.0"encoding="UTF-8"?>
<xsl:stylesheetversion="1.0"xmlns:xsl="http://www.w3.org/199
9/XSL/Transform">
<xsl:outputmethod="text"encoding="UTF-8" />
<xsl:strip-spaceelements="*"/>
<xsl:templatematch="/xjx/cmd[@n='as']">&lt;!DOCTYPE html [
&lt;!ENTITY nbsp "&amp;#160;"&gt;
]&gt;
&lt;html&gt;&lt;body&gt;<xsl:value-
```

```
ofselect="text()"/>&lt;/body&gt;&lt;/html&gt;
</xsl:template>
<xsl:templatematch="/xjx/cmd[@n!='as']"/>
</xsl:stylesheet>
```

Notez la génération d'un DOCTYPE définissant l'entité "nbsp", définie en HTML mais pas en XML.

Pour l'appliquer :

```
$ xsltproc leprogramme.xslt a.xml > b.xml
```

On peut maintenant ouvrir ce fichier dans un navigateur pour vérifier que le document XML est valide. Ce n'est pas le cas :

- Il y a des entités mal formées (voir par exemple l'émission D&CO le dimanche soir sur M6)
- Les blocs JSON contiennent contiennent des tags `<br>` non fermés.

De plus, il serait bien utile d'extraire les données JSON sous forme de XML pour pouvoir continuer notre prototypage avec XSLT. Appellons Perl et le module JSON à la rescousse !

```
$ xsltproc leprogramme.xslt a.xml | perl -MJSON -npe '
if (m@^ *<div id="(data_[^"]*)"
style="display:none;">([^}]*})</div>@) {
($id,$data)=($1,jsonToObj($2));
$data = join("\n", map { qq|<div class="$_">$data->{$_}</div>| } keys
%$data);
$_ = "<div id=\"$id\">$data</div>";
s@<br>@<br/>@g
}
s/&([^;]{8})/&amp;$1/g;' > c.xml
```

Lançons un petit test d'extraction XSLT : nous allons lister les chaînes avec `chaines.xslt`.

```
<?xmlversion="1.0"encoding="UTF-8"?>
<xsl:stylesheetversion="1.0"xmlns:xsl="http://www.w3.org/199
9/XSL/Transform">
<xsl:outputmethod="text"encoding="UTF-8" />
<xsl:templatematch="/html/body/div">
<xsl:value-ofselect="substring(@id, 6)"/>: <xsl:value-
ofselect="div[@class='chaine']/@title"/>
<xsl:text>&#10;</xsl:text>
</xsl:template>
</xsl:stylesheet>
```

Et voici le résultat :

```
$ xsltproc chaines.xslt b.xml
192: TF 1
4: France 2
80: France 3
34: Canal+
47: France 5
111: Arte
118: M6
```

OK. Nous avons le champ libre pour transformer notre document en XMLTV avec `xmltv.xslt` :

```
<?xmlversion="1.0"encoding="UTF-8"?>
<xsl:stylesheetversion="1.0"xmlns:xsl="http://www.w3.org/199
9/XSL/Transform">
<xsl:outputmethod="xml"indent="yes"encoding="UTF-8" />
<xsl:strip-spaceelements="*"/>
<xsl:templatematch="/html">
<tvgenerator-info-name="Télérama → XMLTV"
source-info-
url="http://television.telerama.fr/tele/grille.php"
source-info-name="Télérama">
<xsl:apply-
templatesmode="channel"select="body/div/div[@class='chaine']
" />
<xsl:apply-
templatesmode="programme"select="body/div/div/div/div[contai
ns(@class, 'emission')]/div/div[substring(@id, 0,
6)='data_']" />
</tv>
</xsl:template>
<xsl:templatematch="text()"priority="-1"/>
<xsl:templatemode="channel"match="div">
<channel>
<xsl:attributename="id">
<xsl:value-ofselect="substring(../@id, 6)"/>
```

```
</xsl:attribute>
<display-namelang="fr"><xsl:value-
ofselect="./@title"/></display-name>
<iconwidth="40"height="40">
<xsl:attributename="src">http://icon-
telerama.sdv.fr/tele/imedia/images_chaines_tra/Transparent/4
0x40/ <xsl:value-ofselect="substring(../@id,
6)"/>.gif</xsl:attribute>
</icon>
</channel>
</xsl:template>
<xsl:templatemode="programme"match="div">
<programme>
<xsl:attributename="channel">
<xsl:value-ofselect="div[@class='Id_Chaine']"/>
</xsl:attribute>
<xsl:attributename="start">
<xsl:apply-templatesmode="xmltv-time-from-iso-
8601"select="div[@class='Date_Debut']"/>
</xsl:attribute>
<xsl:attributename="stop">
<xsl:apply-templatesmode="xmltv-time-from-iso-
8601"select="div[@class='Date_Fin']"/>
</xsl:attribute>
<xsl:attributename="showview">
<xsl:value-ofselect="div[@class='ShowViewFr']"/>
</xsl:attribute>
<titlelang="fr">
<xsl:value-ofselect="div[@class='Titre']"/>
</title>
<sub-titlelang="fr">
<xsl:value-ofselect="div[@class='Sous_Titre']"/>
</sub-title>
<desclang="fr">
<xsl:value-ofselect="div[@class='resume_long']"/>
</desc>
<categorylang="fr">
<xsl:value-ofselect="div[@class='Type']"/>
</category>
<lengthunits="seconds">
<xsl:value-ofselect="div[@class='DureeEnSecondes']"/>
</length>
</programme>
</xsl:template>
<xsl:templatemode="xmltv-time-from-iso-8601"match="*|@*">
<xsl:value-
ofselect="concat(substring(.,0,5),substring(.,6,2),substring
(9,2),substring(.,12,2),substring(.,15,2))"/>
</xsl:template>
</xsl:stylesheet>
```

Il n'y a plus qu'à enchaîner toutes les étapes dans un script shell (`tv.sh`) :

```
$ ./tv.sh tv.2008-02-14T23:00:00.xml 2008-02-14T23:00:00
```

Et voici quelques extraits du résultat :

```
<?xmlversion="1.0"encoding="UTF-8"?>
<tvgenerator-info-name="Télérama → XMLTV"source-info-
url="http://television.telerama.fr/tele/grille.php"source-
info-name="Télérama">
<!-- […] -->
<channelid="111">
<display-namelang="fr">Arte</display-name>
<iconwidth="40"height="40"src="http://icon-
telerama.sdv.fr/tele/imedia/images_chaines_tra/Transparent/4
0x40/111.gif"/>
</channel>
<!-- […] -->
<programmechannel="111"start="2008020000"stop="2008020220"sh
owview="2779370">
<titlelang="fr">Arizona Dream</title>
<sub-titlelang="fr"/>
<desclang="fr">Axel, un orphelin de 20 ans, vit davantage
dans un monde rempli de rêves et de poissons volants qu'à
New York, où il habite. Par l'esprit, il se «transporte»
souvent sur la banquise lointaine et ...</desc>
<categorylang="fr">Film</category>
<lengthunits="seconds">8400</length>
</programme>
<!-- […] -->
</tv>
```

Pour visualiser le résultat obtenu sous une forme plus lisible pour un humain, je vous

recommande d'essayer XSLTv. Vous enregistrez le fichier XMLTV sous `tv.xml` dans le répertoire de XSLTv et vous chargez `tv.html` dans votre navigateur. Vous obtenez ceci :

Ce n'est pas visuellement aussi beau que le site Télérama, mais pensez que ce n'est que l'une des multiples utilisations du format XMLTV : ces données peuvent aussi être utilisées par exemple pour piloter un magnétoscope numérique.

L'étape suivante sera de programmer un proxy en PHP, mais je vous en parlerais plus tard.

Fichiers joints :

- `leprogramme.xslt`
- `chaines.xslt`
- `xmltv.xslt`
- `tv.sh`

MàJ 2008-03-02 : J'ai mis à jour tv.sh et xmltv.xslt. Voir ce billet.

Commentaires

1. Le samedi 14 juin 2008, 18:54 par weetabix

trop top !!
je vais tester dans la semaine !

2. Le vendredi 25 juillet 2008, 12:24 par yoyo

marche bien et rapide
comment faire pour récupérer les programme sur une semaine entière et pas seulement les 3-4h de l'heure courante ?

3. Le mardi 9 septembre 2008, 12:51 par mod

je voudrais savoir comment connaitre le code showiew de certain programmes afin de pouvoir les enregistrer

4. Le mardi 23 septembre 2008, 14:54 par Olivier - Code & rando

@mod

Le code showview est disponible dans les informations détaillées sur un programme sur le site de Télérama.

Et elles sont également disponibles dans le fichier XMLTV que je génère.

Ajouter un commentaire

Nom ou pseudo :

Adresse email :

Site web (facultatif) :

Commentaire :

Le code HTML est affiché comme du texte et les adresses web sont automatiquement transformées.

PRÉVISUALISER

La discussion continue ailleurs

1. Le dimanche 2 mars 2008, 03:05 par Olivier Mengué – Code & rando

Télérama → XMLTV : mise à jour

2008-03-02 J'ai corrigé les fichiers xmltv.xslt et tv.sh...

URL de rétrolien : http://o.mengue.free.fr/blog/trackback/51

Fil des commentaires de ce billet

Propulsé par Dotclear